Probabilistic Neural Network-Based Sensor Configuration in a Wireless Ad Hoc Network

Thomas J. Stevens
Raytheon Missile Systems / University of Arizona
phone: 520-390-5233
email: tjstevens@raytheon.com

Malur K. Sundareshan
University of Arizona
email: sundareshan@ece.arizona.edu

Abstract We address the following scenario: a single target moves through a field of stationary sensors with known locations. At each time epoch, each sensor is either active or not; each active sensor outputs either target detected or not detected. The probability of target detection is a decreasing function of the distance from a sensor to the target. A particle filter is used to track the target through the sensor field using all active sensor outputs. A probabilistic neural network is used to determine which sensors should be active. The activation function for PNN is the probability of detection for the individual sensor as each node in the hidden layer directly represents one of the sensors in the field. The input to the PNN is a modified version of the estimated target state vector; the modification to the state vector is the addition of a confidence term, which describes the confidence required prior to performing the detections for the subsequent epoch. The output of the PNN is a radius about the previous target location estimate within which to activate sensors to achieve the desired confidence. For a given location in the sensor field the overall probability of detection using distributed detection is several orders of magnitude higher than when using a single sensor; this is due to the overlapping probability of detection regions for the various sensors. Monte Carlo simulations show that the configuration strategy leads to a significant (averaging 30%) reduction in the required number of active sensors with little degradation in the tracker performance. The estimation is performed using a particle filter.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
20 DEC 2004	**N/A**	**-**

4. TITLE AND SUBTITLE	
Probabilistic Neural Network-Based Sensor Configuration in a Wireless Ad Hoc Network	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Raytheon Missile Systems; Department of Electrical and Computer Engineering University of Arizona, Tucson, AZ 85721-0104**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
See also, ADM001741 Proceedings of the Twelfth Annual Adaptive Sensor Array Processing Workshop, 16-18 March 2004 (ASAP-12, Volume 1)., The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**UU**	**20**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

PROBABILISTIC NEURAL NETWORK-BASED SENSOR CONFIGURATION MANAGEMENT IN A WIRELESS *AD-HOC* NETWORK

Thomas J Stevens and Malur K. Sundareshan
Department of Electrical and Computer Engineering
University of Arizona, Tucson, AZ 85721-0104
Tel: (502) 621-2953; e-mail: stevenst@ece.arizona.edu, sundareshan@ece.arizona.edu

ABSTRACT

This paper describes a novel application of a probabilistic neural network for overcoming the computational complexity involved in performing sensor configuration management in a collaborative sensor network. We consider the problem of reliably tracking a target moving through a field of stationary sensors by fusing the measurements returned from the distributed array of sensors while conserving power by minimizing the number of sensors participating in the decision-making at each step, which is a challenging problem of significant current interest. The twin, and often conflicting, requirements of high tracking accuracy (achievable by recruiting more sensors in order to develop fused decisions) and minimization of network latency (performing decisions using measurements from only a small subset of sensors) place a major emphasis on developing optimal strategies for sensor configuration management in such application scenarios. Recently suggested approaches to this problem typically employ Bayesian Networks and Influence Diagrams, which are computationally intensive and are often prohibitive for real time applications, particularly when the number of sensors involved is large. To overcome the computational complexity, we propose the use of a probabilistic neural network (PNN). The task for the PNN is to produce a distance measure (a radius, for instance) about a target location estimate within which to query sensors for observations by using the previous state estimate of the target as input. By integrating the PNN with a particle filter implementation of a tracking algorithm, we develop a collaborative distributed tracking scheme. Performance evaluation results are presented to demonstrate the benefits from sensor fusion (improvement of tracking accuracy) and reduction of latency (saving in the number of sensors deployed for accomplishing the task) in chosen tracking scenarios.

I. Introduction

The availability of efficient, and rather inexpensive, sensors that can be tuned to a wide range of operating conditions enables a multitude of these sensors to be deployed in an array or in multiple arrays to cover a large area under surveillance. For collaborative decision-making for the detection, discrimination, localization, and tracking of targets of interest, these sensors need to be networked for exchange of either raw measurements or some decisions resulting from processing the data, or for exchange of information with a centralized monitoring station at a remote location. Sensors integrated with microprocessors and radios are able to communicate wirelessly as connected computing nodes. Such nodes are capable of making observations about the environment, and by communicating with other nodes work collaboratively to analyze the data. The wireless nature of these *ad hoc* networks allows for widely distributed and massively parallel systems, which only require limited processing ability at the individual nodes. Due to the possibility of collaboration, these systems are capable of resulting in significantly increased overall processing ability while ensuring improved detection probability and tracking accuracy and reducing the probability of false alarms.

The use of sensor fusion methods for collaborative decision-making has been well appreciated [1]. In a typical sensor fusion scheme, measurements from multiple sensors are integrated in order to achieve a common goal and obtain an overall performance that is better than that could result from individual sensors acting alone. Depending on the architecture employed for the integration, fusion can be performed at the data level (combining raw data output by the sensors), or at the feature level (combining of features extracted from each data stream), or at the decision level (combining of final decisions made from exploiting data measured by each sensor). Although the collaborative processing in a sensor network also involves fusing data from various sensors, what makes this process different from traditional sensor fusion methods is the *ad hoc* nature of the network, *viz.* the collaborating partners for each sensor can be varied from one instant to another as decisions are made. Furthermore, the signal processing steps required to produce any decision (which may include ordering the data from different sensors, extracting features, fusing the features, and computing the decision) can all be performed at any given sensor at a given instant, as distinct from using a fixed architecture for fusion where most of the signal processing takes place at a centralized location which receives the measurements from the various sensors. It is the distributed nature of sensor placement coupled with the dynamically varying locations at which sensor fusion and decision-making functions take place that makes the operation of a collaborative sensor network significantly more challenging. The crux of the problem is to devise an intelligent scheme for “sensor configuration management” that provides a mechanism for each sensor needing to make a decision to recruit collaborating partners within its neighborhood in a dynamic manner.

A major practical concern in the employment of *ad hoc* sensor networks is to constrain the average power required by the network to perform the needed task, which requires limiting the number of active sensors at any given time. Unfortunately, however, reducing the number of sensor nodes could result in a reduction of the surveillance and tracking accuracy, thus requiring intelligent approaches to minimize the “network latency”. The twin goals of achieving improved performance by sensor fusion and minimizing network latency (*i.e*. ensuring only limited degradation in overall performance due to employing only a subset of available sensors) make sensor configuration management particularly challenging.

Some recently developed approaches to address the configuration management problem propose use of Bayesian Networks, which facilitate incorporation of expert knowledge in the configuration process [2]. This formalism allows one to construct an Influence Diagram to evaluate the utility of a decision made at each sensor node. The predicted utility of a particular decision is then used to obtain a network configuration from the perspective of the individual decision-making node. For addressing the specific problem of surveillance and tracking, one can then employ an optimization framework in order to minimize the predicted error while keeping the number of sensors low. Unfortunately however, the use of influence diagrams is computationally intensive and is often prohibitive for real time applications, particularly when the number of sensors involved is large.

The computational complexities involved in the configuration management can be effectively addressed by using a probabilistic neural network (PNN), as will be shown in this paper. The approach used here exploits the implied correspondence that exists between a Bayesian Network model and that provided by the PNN. While the use of a PNN by itself to perform sensor configuration management may be quite demanding as the PNN requires training prior to implementation, combining the Bayesian Network and PNN models has significant advantages, a feature that will be exploited in the present work. In particular, by combining these methods one is able to significantly reduce the prior training required for the PNN while avoiding the intensive computation that may be required for obtaining inference from the Bayesian Network of the probability that a particular configuration of sensor nodes will provide an accurate target state estimate. For facilitating developmental details and to serve as a vehicle for quantitative performance evaluation, the specific problem of tracking a vehicle entering a sensor field will be addressed in this paper. The task for the PNN as used here is to produce a distance measure (a radius, for instance) about a target location estimate within which to query sensors for observations by using the previous state estimate of the target as input. Performance evaluation results will be presented to demonstrate the benefits from sensor fusion (improvement of tracking accuracy) and reduction of latency (saving in the number of sensors deployed for accomplishing task) in chosen tracking scenarios.

II. Description of Tracking Scenario

To perform tracking of a moving object in a sensor field, global information in both space and time must be collected and analyzed over a time horizon and over a spatial region. Each individual node however can provide only spatially local information. Furthermore, due to power limitation, temporal processing is feasible over only small time intervals. This requires collaborative processing of collected information. For collaborative decision-making to achieve optimal tracking performance, we consider the following scenario. In the plane in which the target moves, there are N sensors uniformly placed whose locations are known. At each time, each sensor may be configured to be active or inactive. Active sensors collect M samples of a received signal. Each sensor node that detects an event of interest is capable of running independently the needed signal processing algorithms in order to maintain contact with the target. However, for improving its own performance, each sensor will have the ability to query its neighbors for their assessment of the situation. Upon receipt of assessments from the queried neighbors, it updates its own decision, which will in turn be reported to the neighbors or to a remote monitoring station.

The schematic for the overall tracking scheme used in this paper is shown in Figure 1. The tracking process begins with all sensors making observations. If some sensor has detected a target, then all the observations are reported to the tracking filter that is configured as a particle filter [3]. The particle filter, which performs Bayesian state estimation, is initialized with a uniform prior probability distribution. The particles are then re-sampled according to the observations presented.

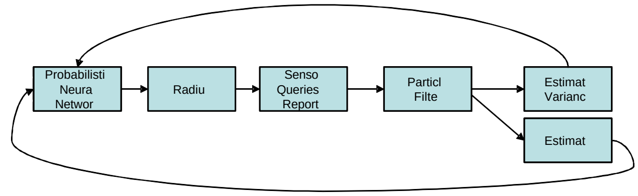


Figure 1 - Block Diagram of Collaborative Tracker.

As noted earlier, our objective in sensor configuration is to accurately track the target through the distributed sensor network while at the same time minimize the number of active sensors (only active sensors are configured to provide observations, else they go to a sleep mode to conserve power). Consequently, activation of sensors within a specified distance from the current target position estimate is of importance. Several interesting formulations of this problem are possible in order to develop the needed tools for ensuring a seamless determination of the “sensor activation region” as the target of interest moves within the field under surveillance. These range from simple heuristic procedures, in which sensors within a specified distance from the estimated target position are queried (with the distance chosen adaptively based on the accuracy of the target position estimate) to more sophisticated procedures that involve solving an optimization problem for a “sensor configuration parameter” (say, the radius of a spherical region). Since keeping up with fast moving objects is of interest for real-time implementation, inclusion of procedures that enable breaking the computational complexity is of particular interest. A closely related problem within the overall sensor configuration management is the development of needed protocols for the “transfer of leadership” from one sensor to another in a “moving leader strategy” in order to continue tracking with the same efficiency as the target moves from one region to another. This problem will not be addressed in this paper.

2.1 Models for Target Motion and Sensor Observation

The tracking scenario consists of a single target moving through a field of stationary sensors with known locations. At each time epoch, each sensor is either active or not; each active sensor outputs either “target detected” or “target not detected”. The probability of target detection is a decreasing function of the distance from a sensor to the target. A target state estimation scheme is used to track the target through the sensor field using all active sensor outputs. For designing the target state estimator, a simple motion model described below will be used.

The target is constrained to motion in a plane. The target state is modeled using position and velocity at time k as measured in Cartesian coordinates. A discrete-time linear system driven by white Gaussian noise is used to model the target dynamics [4]. We arrange the target position, r, and the target velocity, c, as vectors in Cartesian coordinates $r[k]$ and $c[k]$. Using these vectors, we create a target state vector x_{k} defined by

$$\mathrm{x}_{\mathrm{k}} = \begin{bmatrix} r[k] \\ c[k] \end{bmatrix} \quad (1)$$

The discrete-time system model represents snapshots of position and velocity, *which evolve continuously* at evenly spaced instants of time t_0, t_1, etc.; spaced ? time units apart. The system dynamics are given by

$$\mathrm{x}_{\mathrm{k+1}} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathrm{x}_{\mathrm{k}} + \mathrm{w}_{\mathrm{k}} = A\mathrm{x}_{\mathrm{k}} + \mathrm{w}_{\mathrm{k}} \quad (2)$$

where w_{k} is a vector white Gaussian noise process with constant covariance matrix Q .

The sensor receives energy from the target which is then compared to a threshold to determine that an observation has been made. We denote the energy per sample of the signal received from the target as $s_T^2(d)$. This energy is inversely proportional to the square of the distance d between the target and the sensor, *i.e.* $s_T^2(d) = \frac{s_{T_0}^2}{d}$, where $s_{T_0}^2$ is the energy per sample of the target signal at a distance of 1 unit. The energy per sample of the noise at each sensor is denoted by s_N^2 .

In the tracking scenario considered here, N sensors are at known locations after being uniformly placed in the region. The model for sensor observations [5] is given by the probability of detection and the probability for false alarm, assumed in the form

$$P_{FA} = \frac{\Gamma\left(\frac{M}{2}, \frac{b}{2}\right)}{\Gamma\left(\frac{M}{2}\right)} \quad (3)$$

and

$$P_D(d) = \frac{\Gamma\left(\frac{M}{2}, \frac{b}{2}\frac{1}{1+\frac{SNR_0}{d^2}}\right)}{\Gamma\left(\frac{M}{2}\right)} \quad (4)$$

where $SNR_0 = \frac{s^2_{T_0}}{s^2_N}$ is the signal to noise ratio at unit distance from sensor to target, b is a parameter chosen to give the desired probability of false alarm, and $\Gamma(\cdot,\cdot)$ is the incomplete gamma function. Figure 2 shows a typical plot of the probability of detection $P_D(d)$ as a function of distance *d*.

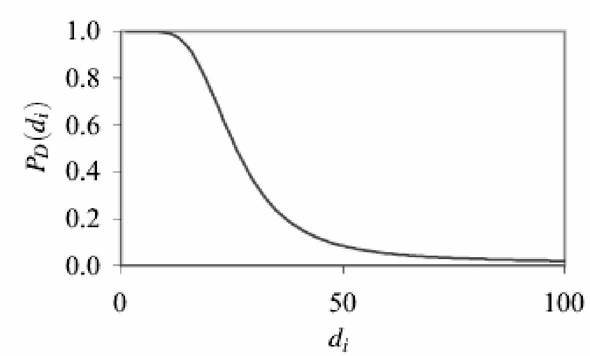


Figure 2 - Probability of detection as a function of target-sensor distance.

2.2 Particle Filter for Tracking

The primary goal of the tracking algorithm is to provide the estimate of target state by processing the sensor measurements. Each sensor activated by the sensor configuration management algorithm independently produces observations, which are then integrated into the global position estimate of the target produced by the tracking algorithm. While several alternate approaches (such as Kalman filters [6], extended Kalman filters, neural network-based nonlinear estimation schemes [7]) are possible for designing systematic tracking algorithms, we select an implementation based on a Particle Filter (PF) for the present application. The PF implements a sequential Monte-Carlo estimation procedure [3] based on point-mass (or particle) representations of probability densities by attempting to compute a sampled representation of the probability distribution, $p(x_k \mid x_{k-1})$, as given by the system dynamics in Eq. (2).

A set of arbitrarily selected particles, $\{x_k^{(j)}\}$, each of which forms an independent hypothesis of the target state at a given time, *k*, are selected, and a set of weights, $\{w_k^{(j)}\}$, computed from the observations returned from sensors, is used to weight the particles to provide an approximation to the posterior probability distribution of the target state. The PF sequentially updates the particles as time progresses and correspondingly updates the set of weights. The state estimate, $\hat{x}_{k-1}$, and its associated covariance matrix, $P_{k|k}$, are then computed using

$$\hat{x}_{k|k} = \sum_j w_k^{(j)} \cdot x_k^{(j)} \qquad (5)$$

$$P_{k|k} = \sum_j w_k^{(j)} \left(x_k^{(j)} - \hat{x}_{k|k}\right)\left(x_k^{(j)} - \hat{x}_{k|k}\right)^T \qquad (6)$$

Since a weighted averaging is performed, selection of a large set of particles in effect contributes to a good approximation to the posterior distribution function. However, this also increases the computational burden. More details on the state estimation properties of PFs are omitted due to page restrictions; they may however be found in [9].

III. Probabilistic Neural Network for Configuration Management

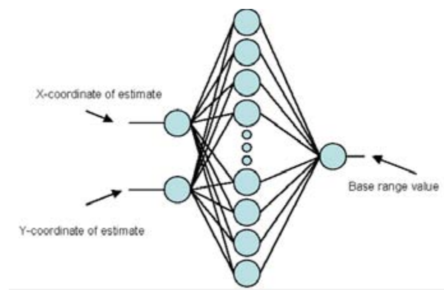


Figure 3 - Structure of PNN

The task for the PNN is to identify a "sensor activation region" given the current position estimate of the target in order to determine which sensors should be active. The neural network is configured with a hidden layer node for each sensor node in the sensor field and is initialized with all connections weights set to unity. The variance of the initial estimate is then used to modify the connection weights. The activation function of the hidden layer nodes is the probability of detection curve for the associated sensor as given in Eq. (4) and has a threshold of 70%, arbitrarily chosen. As each node in the hidden layer directly represents one of the sensors in the field, the network then represents a PDF for the probability of detection for the entire region. One may refer to [8] for details on the properties of PNNs.

The input to the PNN is a modified version of the estimated target estimate,

$$\hat{x}_{k_{inp}} = \hat{r}[k] \cdot P_{D_{req}} \cdot N \cdot s_{P_{k|k}} \qquad (7)$$

where $\hat{r}[k]$ is the prior target position estimate, $P_{D_{req}}$ is the required probability of detection value for performing the detections during the subsequent epoch, N is a normalizing factor and $s_{P_{k|k}}$ is the variance of the covariance matrix for the prior estimate. The use of $s_{P_{k|k}}$ in computing the input to PNN provides a mechanism for taking into account the quality of the prior estimate. Adjusting the connection weights in this way has the effect of modifying the threshold of the associated hidden layer nodes.

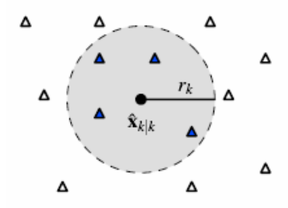


Figure 4 - Example of sensor activation region about previous target state estimate.

The output of the PNN is a radius parameter, r_k, specifying the activation region around the previous target location estimate, $\hat{x}_{k|k}$, within which to activate sensors in order to achieve the desired probability of detection. This parameter bounds the maximum distance of an activated node to the current target position estimate and is used as the basis for querying sensors for the subsequent epoch. In this way, the hidden layer nodes of the PNN represent the effective probability of detection of a target given a previous state estimate and the current sensor configuration, represented by $P_d(x \mid \hat{x}_{k-1}, c_k)$, where x is the target location, $\hat{x}_{k-1}$ denotes the previous state estimate and c_k denotes the current network configuration. Figure 5 depicts the effective overall P_d for the case of uniform sensor layout.

(a) (b)

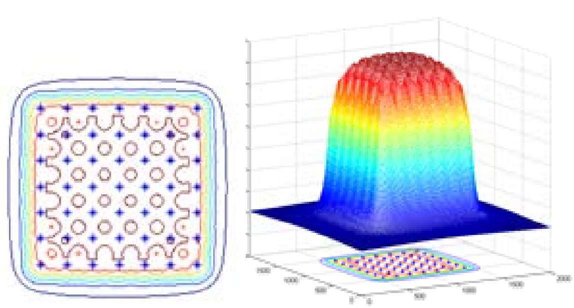


Figure 5 – (a) Contour and (b) Volumetric plots of the overall PDF for Pd created by PNN.

IV. Performance Evaluation

The performance of the present configuration management scheme was evaluated using two sets of 500 simulation runs: one set with all sensors active and the other set by implementing the present configuration management strategy. The target was tracked by an array of 50 sensors distributed in a 1km square area. For providing ground truth to test the estimator quality, a true target trajectory was generated for 25 time steps. In each set of 500 simulation runs, the target track was estimated from simulated sensor detections in the two cases (i) with all sensors enabled, and (ii) with only those sensors within the activation regions provided by the PNN.

For a given location within the sensor field, the overall probability of detection resulting from the collaborative distributed detection scheme is several orders of magnitude higher than when using a single sensor; this is due to the overlapping probability of detection regions for the various sensors. Monte Carlo simulations show that the configuration management strategy leads to a significant (averaging 30%) reduction in the required number of active sensors with little degradation in the tracker performance. The sensor locations, the true target track, and the mean estimated tracks obtained with configuration-managed (PNN activated sensors) and unmanaged (all sensors active) Monte Carlo runs (500 iterations) are shown in Figure 6.

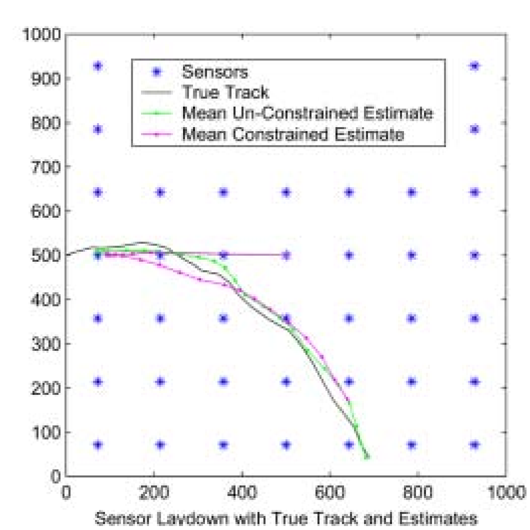


Figure 6 - Sensor lay down with Track and Estimates

The mean squared error computed in both cases (configuration-managed and unmanaged) of simulation runs are shown in Figure 7. A histogram of the mean number of sensors used in each of the 500 runs of the configuration-managed tracking scheme is shown in Figure 8. As can be seen, the PNN configuration management strategy offers a significant reduction in the number of sensors used. Tracking with sensors selected by the PNN on average requires less than 30% of the total sensors deployed while resulting only in a marginal degradation of the tracking accuracy. As may be observed from Figure 7, there is only a slight increase in tracking errors (less than 10% increase on average).

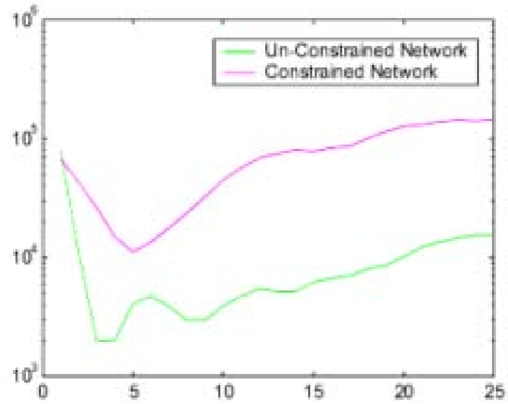


Figure 7 - Mean Square Error from 500 Monte Carlo runs.

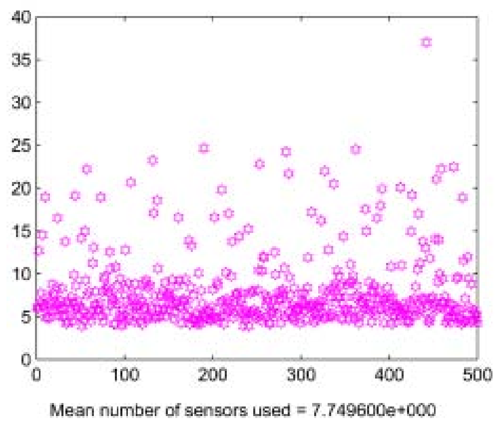


Figure 8 - Histogram of activated sensors

V. Conclusions

The major contribution of this paper is a novel approach to sensor configuration management using probabilistic neural networks. The present strategy enables real-time implementation of *ad hoc* sensor networks operating in power-constrained environments for collaborative surveillance and tracking. Performance evaluation results presented here show that the number of activated sensors can be dramatically reduced without significant increases in tracking error. Planned work for the future will extend the target tracking and sensor configuration algorithms to scenarios with multiple targets and diverse sensors.

REFERENCES

[1] J. Manyika and H. Durrant-Whyte, *Data Fusion and Sensor Management: A Decentralized Information-Theoretic Approach*, Ellis Horwood: New York, 1994.

[2] F. Jensen, *Bayesian Networks and Decision Graphs*, Springer-Verlag, 2002.

[3] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking", *IEEE trans. on Signal Processing*, Vol. 50, pp 174-188, 2002.

[4] T. Stevens and D. Morrell, "Minimization of sensor usage for target tracking in a network of irregularly spaced sensors", *Proc. 2003 IEEE Workshop on Statistical Signal Processing*, St. Louis, MO, 2003.

[5] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice*, Prentice-Hall:New Jersey, 1993.

[6] M. K. Sundareshan and F. Amoozegar, "Data fusion and nonlinear tracking filter implementation using multilayer neural networks," *Proc of IEEE Int. Conf. on Neural Networks (ICNN'95)*, Perth, Australia, November 1995.

[7] W. H. Hsu, "A position paper on statistical inference techniques which integrate neural network and Bayesian network models", *Proc. of 1997 Int. Conf. on Neural Networks*, Vol.3, pp 1972 – 1977, June 1997.

[8] D. Specht, "Probabilistic Neural Networks", *Neural Networks*, Vol. 3, pp. 109-118, 1990.

[9] A. Doucet, et al., *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, New York, NY, 2001, pp. 17-38

[10] D. Specht, "Probabilistic Neural Networks for Classification, Mapping, or Associative Memory", Research Report, Lockheed Palo Alto Research Laboratories, Palo Alto, CA 94304, 1989.

[11] M. B. Menhaj, and F. Delgosha, "A soft probabilistic neural network for implementation of Bayesian classifiers", *Proc. of Int. Joint Conf. on Neural Networks* (IJCNN '01), Vol.1, pp. 454 - 458 July 2001.

[12] S. Watanabe, "Algebraic analysis for neural network learning", *Proc. of IEEE Int. Conf. on Systems, Man, and Cybernetics*, Vol.1, pp 431-436, Oct. 1999.

[13] H. Pan, Z. P. Liang, T. J. Anastasio, and Huang, T.S., "A hybrid NN-Bayesian architecture for information fusion", ICIP 98. *Proc. of 1998 International Conference on Image Processing* (ICIP 98), Vol.1, pp 368 – 371, Oct. 1998.

[14] P. Myllymaki, and H. Tirri, "Learning in neural networks with Bayesian prototypes", *Southcon/94. Conference Record*, 29-31, pp 60 – 64, March 1994.

[15] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration for tracking applications", *IEEE Signal Processing Magazine*, Vol. 19, pp. 61–72, March 2002.

PROBABILISTIC NEURAL NETWORK-BASED SENSOR CONFIGURATION MANAGEMENT IN A WIRELESS AD-HOC NETWORK

Thomas J Stevens and Malur K. Sundareshan
Department of Electrical and Computer Engineering
University of Arizona, Tucson, AZ 85721-0104
Tel: (502) 621-2953; e-mail: stevenst@ece.arizona.edu, sundareshan@ece.arizona.edu

Problem of Interest

- In a Collaborative Sensor Network, Critical Problem is to Optimally Manage Sensors
  - Multitude of Sensors Deployed
  - In Arrays or Multiple Arrays
- Optimal Strategies for Sensor Configuration Management Needed
  - Maximize Performance from Fusion (Collaborative Decision-making)
  - Minimize Network Latency

    (due to constraints on power and communication bandwidth)
- Basic Problem
  - Each sensor needs to dynamically make intelligent decisions to recruit collaborating partners within its neighborhood
- Typical Solutions for Configuration Management
  - Bayesian Networks, Influence Diagrams
  - Computation of “Utility” of Decision Made at Each Sensor Node.
- Computational Intensity is a Major Bottleneck
  - Real-time Implementation is Principal Requirement
- How to Break Computational Complexity?
  - Use of Trained Probabilistic Neural Network
  - Reduction of Training Effort from Exploiting Correspondence with Bayesian Network Models

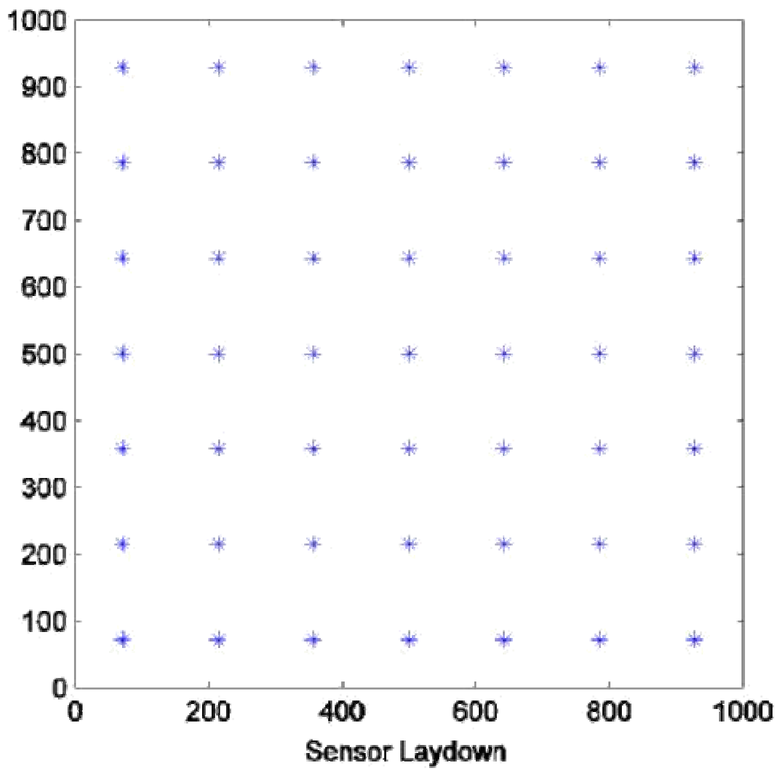
Description of Tracking Scenario

Tracking Moving Object in a Sensor Field
- Requires Global Information in Both Space and Time.
- Info. Analysis Over a Time Horizon and Over a Spatial Region

- Problem Constraints
  - Individual Sensor Nodes Only Provide Spatially Local Information
  - Limited Temporal Processing (due to power constraints)

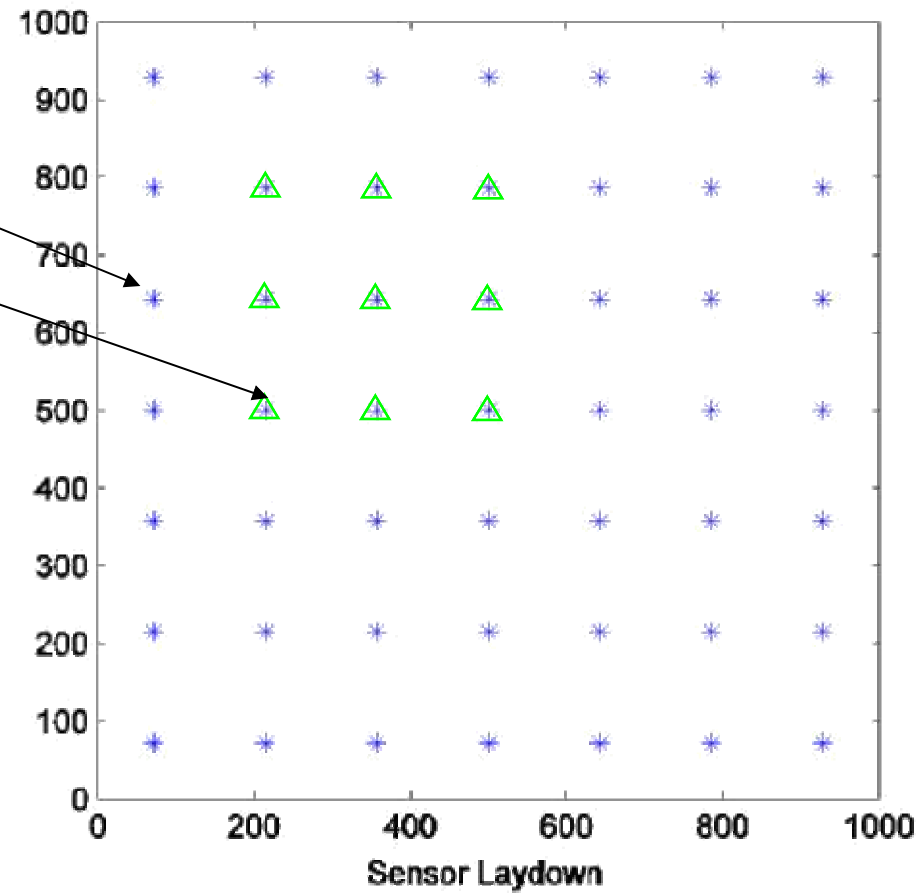
- Needs for Intelligent Solution
  - Collaborative Processing
  - Fusion of Collected Information
  - Collaborative Decision-making

- *N* Sensors Uniformly Placed at Known Locations



Description of Tracking Scenario

- Sensor Modes
  - At Each Time Epoch, Individual Sensor Configured "Active" or "Inactive"
- Active Sensors
  - Collect *M* Samples of Received Signal.
  - Independently Maintain Contact with Target
- Sensor Capabilities
  - Query Neighbors
  - Compute and Update Own Decisions
  - Communicate Wirelessly with Neighbors or with Remote Monitoring Station.



Target Motion Model

- Target State At Time *K*
  - Position
  - Velocity

  (in Cartesian Coordinates)

$$\mathrm{x}_{\mathrm{k}} = \begin{bmatrix} r[k] \\ c[k] \end{bmatrix}$$

- A Discrete-time Linear System Driven By White Gaussian Noise

$$\mathrm{x}_{\mathrm{k+1}} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathrm{x}_{\mathrm{k}} + \mathrm{w}_{\mathrm{k}} = A\mathrm{x}_{\mathrm{k}} + \mathrm{w}_{\mathrm{k}}$$

Sensor Observation Model

- Probability of Detection
  - $\Gamma(\cdot,\cdot)$ Is the Incomplete Gamma Function

$$P_D(d) = \frac{\Gamma\left(\frac{M}{2}, \frac{\beta}{2}\frac{1}{1+\frac{SNR_0}{d^2}}\right)}{\Gamma\left(\frac{M}{2}\right)}$$

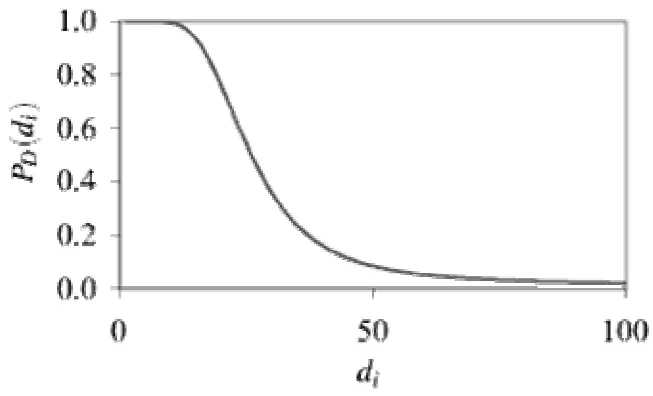
- Probability for False Alarm
  - β Gives Desired Probability of False Alarm

$$P_{FA} = \frac{\Gamma\left(\frac{M}{2}, \frac{\beta}{2}\right)}{\Gamma\left(\frac{M}{2}\right)}$$

- Signal to Noise Ratio at Unit Distance From Sensor to Target

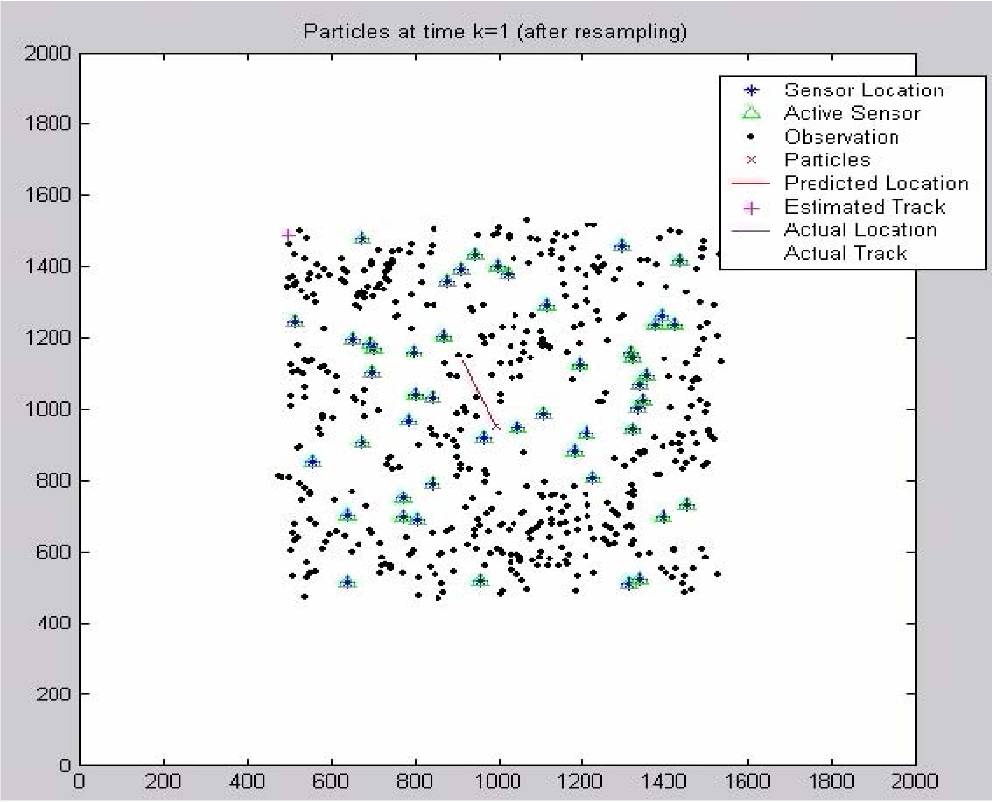
$$SNR_0 = \frac{\sigma_{T_0}^2}{\sigma_N^2}$$

- Typical Plot of Probability of Detection as Function of Distance *D*.



Description of Tracking Filter

- Tracking Begins With All Sensors Making Observations
  - All Observations Reported To Tracking Filter



- Some Sensor Detects Target
- Particle Filter
  - Performs Bayesian Estimation
  - Initialized With Uniform Prior
- Particles are Re-sampled According to Observations
- Sensor Minimization from Configuration Management

Particle Filter Details

- Computes Sampled Representation of Probability Distribution
- Weighted Particles Provide Approximation to Posterior Distribution of Target State
- Sequential Update at Each Time Epoch
  - Distribution of Particles According to Prior
  - Compute Weights According To Distance from Current Observations
- State Estimate Obtained as Weighted Sum of Particles
- Variance of State Estimate Provides Measure of Quality of Estimate

$$p(x_k \mid x_{k-1})$$

$$\left\{x_k^{(j)}\right\}$$

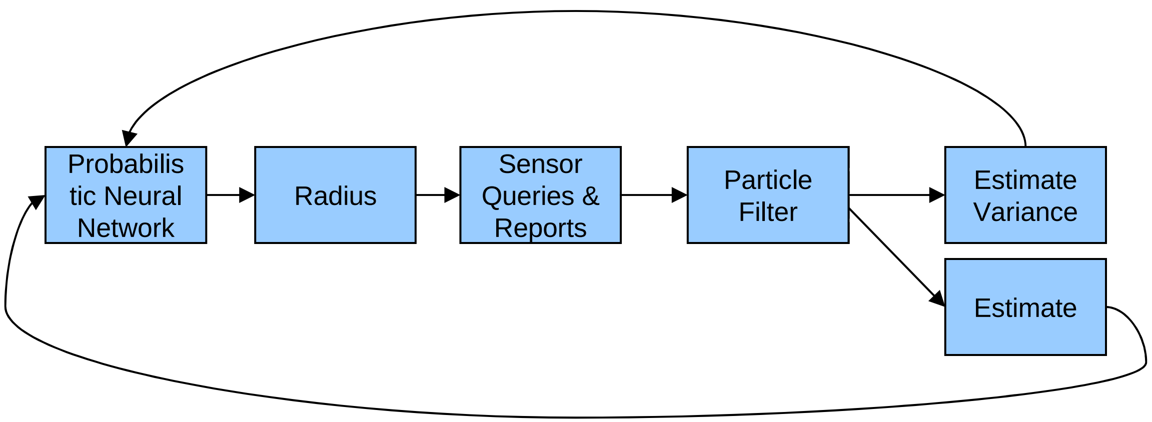
$$\left\{w_k^{(j)}\right\}$$

$$\hat{x}_{k|k} = \sum_j w_k^{(j)} \cdot x_k^{(j)}$$

$$P_{k|k} = \sum_j w_k^{(j)} \left(x_k^{(j)} - \hat{x}_{k|k}\right)\left(x_k^{(j)} - \hat{x}_{k|k}\right)^T$$

Probabilistic Neural Network for Configuration Management

- Tracking Begins with All Activated Sensors Making Observations
- Observations are Reported to Tracking Filter
  - Initialized With A Uniform Prior Distribution
- Particles are Re-sampled According to Observations Presented
- Estimate and Variance are Calculated
  - Used By PNN For Sensor Configuration



Probabilistic Neural Network Configuration

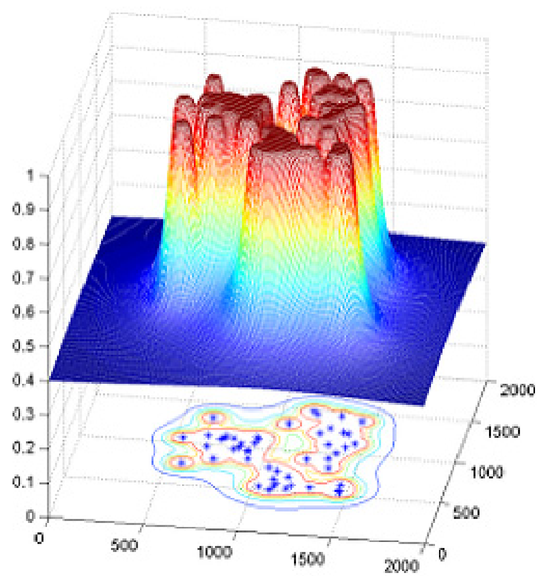
- Input to PNN
  - Modified State Estimate
- Variance Used In Training
  - Provides For Quality of Estimate
- Modifying Hidden Layer Thresholds
  - Generates P_d Surface
  - Accomplished by Adjusting Connection Weights With Variance

$$\hat{x}_{k_{inp}} = \hat{r}[k] \cdot P_{D_{req}}$$

- $\hat{r}[k]$ Prior Target Position Estimate
- $P_{D_{req}}$ Desired Probability Of Detection

$$w_k^j = w_{k-1}^j + N \cdot \sigma_{P_{k|k}} \cdot w_{k-1}^j$$

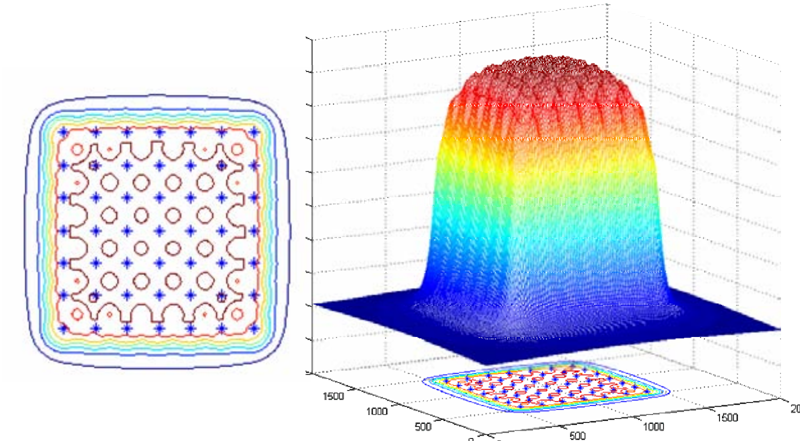
- N Normalizing Factor
- $\sigma_{P_{k|k}}$ Variance Of The Prior Estimate



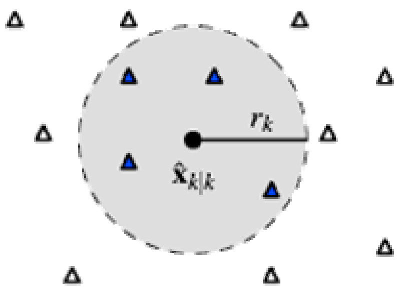
Initial PNN Probability Surface That a Target Will be Detected When All Sensors are Activated – Unit Connections, Random Sensor Lay-down

PNN Output

- Radius Parameter r_k
  - Specifies Activation Region
- Centers On Previous Target Location Estimate $\hat{x}_{k|k}$
- Activates Sensors In Region
  - Region Scaled to achieve Desired P_d
- Bounds Maximum Activation Distance From The Current State Estimate
- Used as Basis for Querying Sensors over Subsequent Epoch.

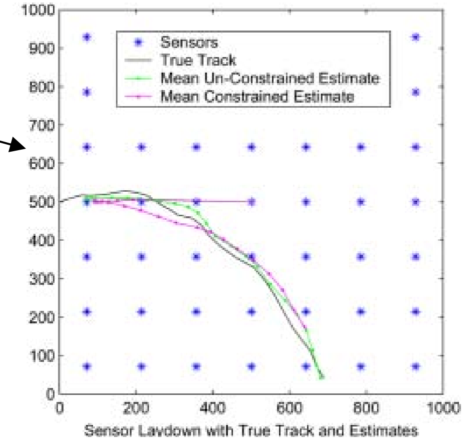


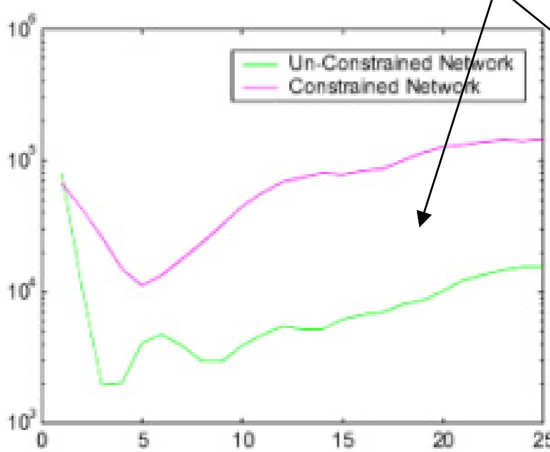
Initial PNN Probability Surface That A Target Will Be Detected When All Sensors Are Activated – Unit Connections, Uniform Sensor Lay-down



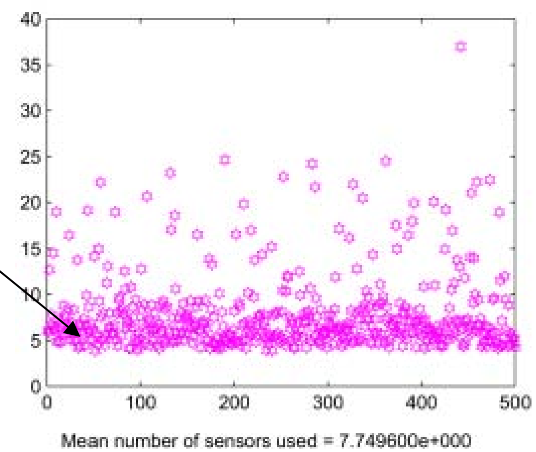
Demonstration of Performance

- Tracking Results – 50 Distributed Sensors in 1km Square Area
- 500 Iterations in Both Cases
  - All Sensors Activated
  - PNN Configuration Managed
- Only Marginal Increase in MSE
- ~1/6th Of Sensors Used





Mean Square Error From 500 Monte Carlo Runs



Conclusions

- Major Contribution
  - Novel Approach to Sensor Configuration Management Using Probabilistic Neural Networks
- Present Strategy Enables Real-time Implementation of *Ad Hoc* Sensor Networks
  - Operating in Power-constrained Environments
  - Collaborative Surveillance and Tracking
- Performance Evaluation Results Show
  - Number of Activated Sensors can be Dramatically Reduced
  - No Significant Increase in Tracking Error
- Planned Work For Future
  - Extend Target Tracking and Sensor Configuration Algorithms to Scenarios with Multiple Targets and Diverse Sensors